Initial Results & Code Final

McKenzie - #500730860

2023-06-25

##libraries  
library(SmartEDA)

## Registered S3 method overwritten by 'GGally':  
## method from   
## +.gg ggplot2

library(tidygeocoder)

## Warning: package 'tidygeocoder' was built under R version 4.3.1

library(dplyr)

##   
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':  
##   
## filter, lag

## The following objects are masked from 'package:base':  
##   
## intersect, setdiff, setequal, union

library(xts) #Load package

## Loading required package: zoo

##   
## Attaching package: 'zoo'

## The following objects are masked from 'package:base':  
##   
## as.Date, as.Date.numeric

##   
## ######################### Warning from 'xts' package ##########################  
## # #  
## # The dplyr lag() function breaks how base R's lag() function is supposed to #  
## # work, which breaks lag(my\_xts). Calls to lag(my\_xts) that you type or #  
## # source() into this session won't work correctly. #  
## # #  
## # Use stats::lag() to make sure you're not using dplyr::lag(), or you can add #  
## # conflictRules('dplyr', exclude = 'lag') to your .Rprofile to stop #  
## # dplyr from breaking base R's lag() function. #  
## # #  
## # Code in packages is not affected. It's protected by R's namespace mechanism #  
## # Set `options(xts.warn\_dplyr\_breaks\_lag = FALSE)` to suppress this warning. #  
## # #  
## ###############################################################################

##   
## Attaching package: 'xts'

## The following objects are masked from 'package:dplyr':  
##   
## first, last

library(sf)

## Warning: package 'sf' was built under R version 4.3.1

## Linking to GEOS 3.11.2, GDAL 3.6.2, PROJ 9.2.0; sf\_use\_s2() is TRUE

library(ggplot2)  
library(tidyverse)

## Warning: package 'tidyverse' was built under R version 4.3.1

## Warning: package 'readr' was built under R version 4.3.1

## Warning: package 'lubridate' was built under R version 4.3.1

## ── Attaching core tidyverse packages ──────────────────────── tidyverse 2.0.0 ──  
## ✔ forcats 1.0.0 ✔ stringr 1.5.0  
## ✔ lubridate 1.9.2 ✔ tibble 3.2.1  
## ✔ purrr 1.0.1 ✔ tidyr 1.3.0  
## ✔ readr 2.1.4

## ── Conflicts ────────────────────────────────────────── tidyverse\_conflicts() ──  
## ✖ dplyr::filter() masks stats::filter()  
## ✖ xts::first() masks dplyr::first()  
## ✖ dplyr::lag() masks stats::lag()  
## ✖ xts::last() masks dplyr::last()  
## ℹ Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors

library(forecast)

## Warning: package 'forecast' was built under R version 4.3.1

## Registered S3 method overwritten by 'quantmod':  
## method from  
## as.zoo.data.frame zoo

library(rvest)

##   
## Attaching package: 'rvest'  
##   
## The following object is masked from 'package:readr':  
##   
## guess\_encoding

# Contents

### Cleaning and ETL

*Loading and unioning data*

* The shelter data is being pulled from the city of Toronto’s [Daily Shelter & Overnight Service Occuapncy & Capacity](https://open.toronto.ca/dataset/daily-shelter-overnight-service-occupancy-capacity/) dataset on Toronto’s open portal website.
* The dataset is split into 3 files,1 for each year. For the 2023 year file, the data source is updated daily.

##Loading data and cleaning date field for union  
   
shelter2021<-read.csv('C:\\Users\\User\\Desktop\\Big Data Analytics Program\\Project\\Tentative Project Datasets\\Shelter\\Daily Shelter Occupancy\\Current\\Daily shelter overnight occupancy (2021).csv')  
shelter2021$OCCUPANCY\_DATE<-as.Date(shelter2021$OCCUPANCY\_DATE, format = "%y-%m-%d")  
  
shelter2022<-read.csv('C:\\Users\\User\\Desktop\\Big Data Analytics Program\\Project\\Tentative Project Datasets\\Shelter\\Daily Shelter Occupancy\\Current\\Daily shelter overnight occupancy (2022).csv')  
shelter2022$OCCUPANCY\_DATE<-as.Date(shelter2022$OCCUPANCY\_DATE, format = "%y-%m-%d")  
  
shelter2023<-read.csv('C:\\Users\\User\\Desktop\\Big Data Analytics Program\\Project\\Tentative Project Datasets\\Shelter\\Daily Shelter Occupancy\\Current\\Daily shelter overnight occupancy (2023).csv')  
shelter2023$OCCUPANCY\_DATE<-as.Date(shelter2023$OCCUPANCY\_DATE, format = "%Y-%m-%d")  
  
#Union for complete dataset  
shltr<-rbind(shelter2021,shelter2022,shelter2023)

### Dimension Reduction

* The shelter(shltr) data source contains information for two exclusive types of shelter programs. Shelter programs are inclusively either room or bed based. For simplicity reasons of this analysis I wont to keep fields are that inclusive to both shelter types and transform the fields to accommodate both types.
* The following fields will be dropped or transformed.

| Field | Outcome | Reason |
| --- | --- | --- |
| LOCATION\_PROVINCE | Dropped | Single Value: ON |
| OCCUPANCY\_RATE | Calculated Field | Consolidates occupancy rates, inclusive of the capacity type. |
| OVER\_OCCUPIED | Calculated Field | Identifying if a shelter program is over occupied. |
| ADDRESS | Calculated Field | Concatenates LOCATION\_ADDRESS, LOCATION\_CITY, LOCATION\_PROVINCE |

##Creating OVER\_OCCUPIED field. Identifying programs that are over occupied exclusive of the capacity type  
shltr$OVER\_OCCUPIED<-ifelse(shltr$OCCUPANCY\_RATE\_ROOMS==100|shltr$OCCUPANCY\_RATE\_BEDS==100,1,0)  
shltr$OVER\_OCCUPIED<-ifelse(is.na(shltr$OVER\_OCCUPIED)==TRUE,0,1)  
  
##Creating ADDRESS field. It is a program's full address.   
shltr$ADDRESS<-paste(shltr$LOCATION\_ADDRESS,shltr$LOCATION\_CITY,shltr$LOCATION\_PROVINCE)  
  
##OCCUPANCY\_RATES. Creating one field for occupancy rates inclusive of capacity type  
shltr$OCCUPANCY\_RATE<-ifelse(is.na(shltr$OCCUPANCY\_RATE\_BEDS),shltr$OCCUPANCY\_RATE\_ROOMS,shltr$OCCUPANCY\_RATE\_BEDS)  
  
##Drop redundant fields  
shltr<-shltr[,c(1:20,33,34,35)]

*Dealing with Missing Values*

ExpData(shltr,type=2)

## Index Variable\_Name Variable\_Type Sample\_n Missing\_Count  
## 1 1 X\_id integer 121170 0  
## 2 2 OCCUPANCY\_DATE Date 121170 0  
## 3 3 ORGANIZATION\_ID integer 121170 0  
## 4 4 ORGANIZATION\_NAME character 121170 0  
## 5 5 SHELTER\_ID integer 121170 0  
## 6 6 SHELTER\_GROUP character 120954 216  
## 7 7 LOCATION\_ID integer 120724 446  
## 8 8 LOCATION\_NAME character 120249 921  
## 9 9 LOCATION\_ADDRESS character 117652 3518  
## 10 10 LOCATION\_POSTAL\_CODE character 117652 3518  
## 11 11 LOCATION\_CITY character 117638 3532  
## 12 12 LOCATION\_PROVINCE character 117638 3532  
## 13 13 PROGRAM\_ID integer 121170 0  
## 14 14 PROGRAM\_NAME character 121100 70  
## 15 15 SECTOR character 121170 0  
## 16 16 PROGRAM\_MODEL character 121168 2  
## 17 17 OVERNIGHT\_SERVICE\_TYPE character 121168 2  
## 18 18 PROGRAM\_AREA character 121168 2  
## 19 19 SERVICE\_USER\_COUNT integer 121170 0  
## 20 20 CAPACITY\_TYPE character 121170 0  
## 21 21 OVER\_OCCUPIED numeric 121170 0  
## 22 22 ADDRESS character 121170 0  
## 23 23 OCCUPANCY\_RATE numeric 121170 0  
## Per\_of\_Missing No\_of\_distinct\_values  
## 1 0.000 50944  
## 2 0.000 883  
## 3 0.000 35  
## 4 0.000 35  
## 5 0.000 69  
## 6 0.002 69  
## 7 0.004 130  
## 8 0.008 129  
## 9 0.029 124  
## 10 0.029 119  
## 11 0.029 7  
## 12 0.029 2  
## 13 0.000 200  
## 14 0.001 203  
## 15 0.000 5  
## 16 0.000 3  
## 17 0.000 8  
## 18 0.000 5  
## 19 0.000 531  
## 20 0.000 2  
## 21 0.000 2  
## 22 0.000 126  
## 23 0.000 1348

* The proportion of missing values is very small. Most of the missing values are for fields related to location. Since the fields that contain missing values are non-numerical I am going to convert all missing values to “unknown”.

#any missing values for measures?   
anyNA(shltr$OCCUPANCY\_RATE)

## [1] FALSE

anyNA(shltr$OVER\_OCCUPIED)

## [1] FALSE

#The only instances of missing values appear as blanks in location related fields. I am turning all blanks to NAs   
  
shltr[shltr==""]<-"Unknown"  
shltr[is.na(shltr)]<-"Unknown"

#Need to trim leading white spaces from city field  
shltr$LOCATION\_CITY<-trimws(shltr$LOCATION\_CITY)

* Dropping and consolidating redundant fields has taken the data source from 32 fields to 23.

*Additional geographic information*

* Part of my analysis will involve understanding shelter related metrics across different neighborhood.
* The following code brings in location related characteristics. The [census tract boundary file](https://www12.statcan.gc.ca/census-recensement/alternative_alternatif.cfm?l=eng&dispext=zip&teng=lct_000b21a_e.zip&k=%20%20%20%2013089&loc=//www12.statcan.gc.ca/census-recensement/2021/geo/sip-pis/boundary-limites/files-fichiers/lct_000b21a_e.zip) comes from statistics Canada and the city of Toronto’s [Neighbourhood file.](https://open.toronto.ca/dataset/neighbourhoods/)

##Creating a list of addresses for each shelter program. These addresses will be used to bring Census tract/neighbourhoods for each program.   
Addresses<-as.data.frame(unique(paste(shltr$LOCATION\_ADDRESS,shltr$LOCATION\_CITY,shltr$LOCATION\_PROVINCE)))  
#Cleaning Column Name  
colnames(Addresses)<-'Addresses'  
  
#Using the tidygeocoder package to bring in lat and longs to each address for neighbourhood identification  
lat\_longs<-Addresses %>%   
 geocode(Addresses)

## Passing 125 addresses to the Nominatim single address geocoder

## Query completed in: 127.1 seconds

#Removing postal codes that did not return a lat/long  
lat\_longs<-lat\_longs %>%  
 filter(is.na(lat)==FALSE & lat\_longs$Addresses!='')  
#Create a point geometric field using the st package  
lat\_longs<-lat\_longs %>% st\_as\_sf(coords=c('long','lat'))  
lat\_longs<-st\_set\_crs(lat\_longs,4326)  
  
#Raading the census tract boundary file from statistics canada.   
# Provide the link   
  
CT<-read\_sf("C:\\Users\\User\\Desktop\\CTest\\lct\_000a21a\_e.shp")  
#Filter for Ontario Province  
CT<-CT %>% filter(PRUID=='35')  
CT<-st\_transform(CT,crs=4326)  
  
#Toronto Neighbourhoods Profile. To identify a Toronto Neighbourhood map geometric point to the boundary file found on toronto open data portal  
TNei<-read\_sf('C:\\Users\\User\\Desktop\\Big Data Analytics Program\\Project\\Tentative Project Datasets\\Neighbourhoods\\Boundary File\\Neighbourhoods - 4326.shp')  
#Table identifying each CT Id for each shelter  
TorCTs<-st\_join(lat\_longs,CT)  
  
#Decide which fields to keep, bring neighbourhood information to address data source  
TorCTs<-st\_join(TorCTs,TNei)  
TorCTs<-TorCTs[,c(1,15,16)]  
  
##Bring Neighbourhood information to the shltr data set.   
shltr<-left\_join(shltr,TorCTs,by=c('ADDRESS' ='Addresses'))  
  
colnames(shltr)[c(24,25)]<-c('Neighbourhood','Improv\_Status\_Area')

## Exploratory Analysis

* Since the dataset captures information related to shelters each day I need to convert/create time series objects. For the purpose of clearer visualizations I am going to use xts objects as the time-series objects.

*Daily Shelter Intake Total*

* Daily intake totals at the overall level will be the first measure of interest.This will be the main measure used for the initial results and code.

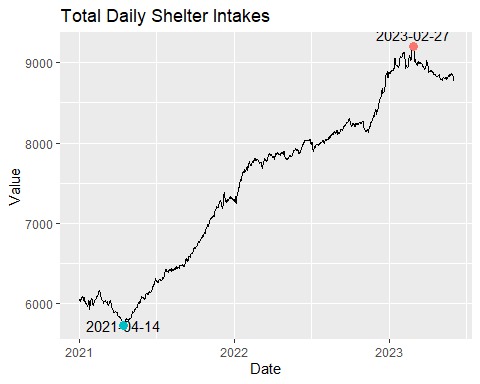
#Defining the index for time for the time series object  
minDate<-min(shltr$OCCUPANCY\_DATE)  
maxDate<-max(shltr$OCCUPANCY\_DATE)  
indy<-seq(minDate,maxDate,by='day')  
  
##Daily Intakes Aggregated for the Toronto Region.   
NTakeShltr<-aggregate(SERVICE\_USER\_COUNT ~ OCCUPANCY\_DATE ,data=shltr,FUN=sum)  
colnames(NTakeShltr)[2]<-'TotalUsers'  
##Creating xts(time series) object  
NTakeShltrVec<-NTakeShltr$TotalUsers  
xtsNTakeShltr<-xts(NTakeShltrVec,order.by = indy)  
colnames(xtsNTakeShltr)<-'Total\_Intakes'

### Toronto’s Daily Shelter Intake Exploratory Analysis

print(summary(xtsNTakeShltr))

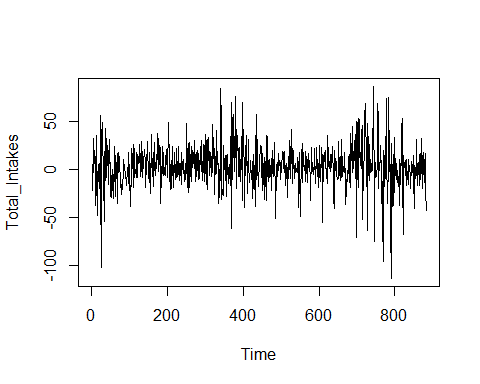
## Index Total\_Intakes   
## Min. :2021-01-01 Min. :5728   
## 1st Qu.:2021-08-09 1st Qu.:6426   
## Median :2022-03-18 Median :7799   
## Mean :2022-03-18 Mean :7498   
## 3rd Qu.:2022-10-24 3rd Qu.:8241   
## Max. :2023-06-02 Max. :9201

# Convert the xts object to a data frame  
TorNtakeShltrPlt <- data.frame(Date = index(xtsNTakeShltr), Value = coredata(xtsNTakeShltr))  
#Creating label to identify dates where we see the maximum shelter intakes  
max\_date <- index(xtsNTakeShltr)[which.max(coredata(xtsNTakeShltr))]  
#Creating label to identify dates where we see the minimum shelter intakes  
min\_date <- index(xtsNTakeShltr)[which.min(coredata(xtsNTakeShltr))]  
  
# Create the plot  
ggplot(data = TorNtakeShltrPlt, aes(x = Date, y = Total\_Intakes)) +  
 geom\_line()+  
#Adding max and min label dates   
 geom\_text(data = subset(TorNtakeShltrPlt, Date %in% c(max\_date, min\_date)),  
 aes(label = as.character(Date), vjust = ifelse(Date == max\_date, -0.5, 0.5)),  
 show.legend = FALSE)+  
#Adding points to id min and max dates   
 geom\_point(data = subset(TorNtakeShltrPlt, Date %in% c(max\_date, min\_date)),  
 aes(color = ifelse(Date == max\_date, "blue",'red')),  
 size = 3)+  
 labs(title = "Total Daily Shelter Intakes") +  
 xlab("Date") +  
 ylab("Value")+  
 theme(legend.position = 'none')



* Looking at the visual we see a increasing trend. This shows that the trend in shelter intakes for the city of Toronto is a non-stationary process. The trend posses qualities of a random walk, there is not a consistent mean or variability. The visual also does not indicate seasonality.
* The lowest levels of intakes occurred on 2021-04-14 with 5728 intakes and the highest level of intakes on 2023-02-27 with 9201 intakes. The mean of the time series is 7498.12 however since this is non-stationary times eries it is not consistent throughout the trend.
* Since we have a random walk time series, it suggest that each observation is a random step or lag from the previous observations. This makes future values unpredictable and suggest that future values depend on current observations.

plot.ts(ts(diff(xtsNTakeShltr)))

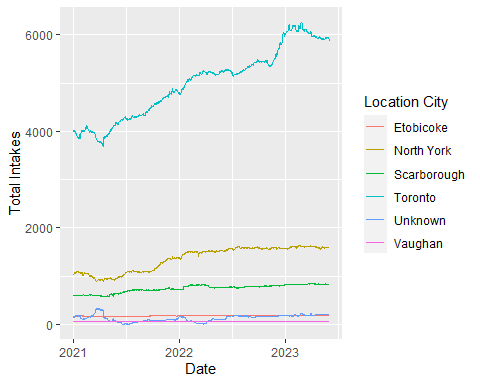


* Differencing (current values - the value directly before it) shows a stationary process.There are some spikes in the variability but we see more variability on the right side of the plot compared to the left.
* Differencing will be important because we need to remove the trend for statistical models.

### Multi-Variate Exploratory Analysis

*City*

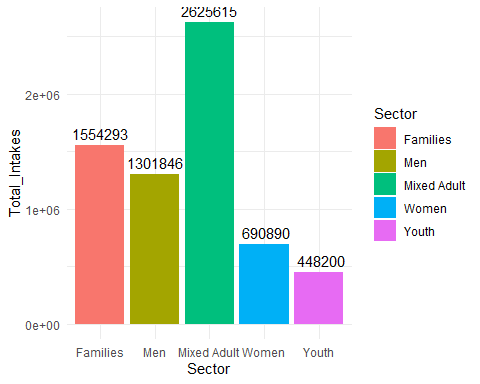
#Pivoting of analysis  
CityNTakeTrend<-as.data.frame(xtabs(data=shltr,shltr$SERVICE\_USER\_COUNT ~ shltr$OCCUPANCY\_DATE + shltr$LOCATION\_CITY))  
colnames(CityNTakeTrend)<-c("OCCUPANCY\_DATE" ,"LOCATION\_CITY","SERVICE\_USER\_COUNT")  
#plot  
ggplot(CityNTakeTrend,aes(x=as.Date(OCCUPANCY\_DATE),y=SERVICE\_USER\_COUNT, color=LOCATION\_CITY))+  
 geom\_line()+  
 labs(x="Date",y="Total Intakes",color="Location City")



* Toronto has the the most intakes consistently It is followed by North York and Scarborough. The trend in the Toronto shelter intakes looks very similar to the overall trend.

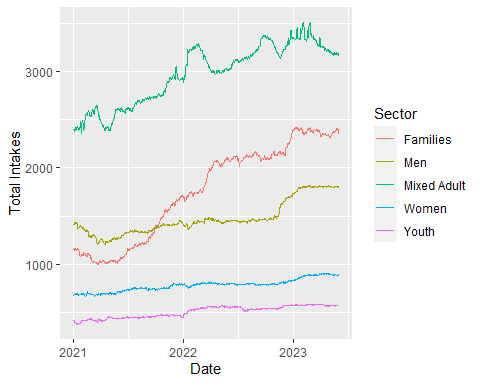
*Sector*

##Total Intakes by Sector  
SecNTake<-shltr %>%  
 group\_by(SECTOR) %>%   
 summarise(TOTAL\_INTAKES = sum(SERVICE\_USER\_COUNT))  
  
ggplot(SecNTake, aes(x = SECTOR, y=TOTAL\_INTAKES,fill=SECTOR))+  
 geom\_bar(stat='identity') +  
 geom\_text(aes(label=format(TOTAL\_INTAKES,scientific=FALSE)),vjust = -0.5 )+  
 labs(x='Sector', y = 'Total\_Intakes', fill='Sector')+  
 theme\_minimal()



* Mixed Adult shelter programs have the most intakes.

#Pivoting of analysis  
SectNTakeTrend<-as.data.frame(xtabs(data=shltr,shltr$SERVICE\_USER\_COUNT ~ shltr$OCCUPANCY\_DATE + shltr$SECTOR))  
colnames(SectNTakeTrend)<-c("OCCUPANCY\_DATE" ,"SECTOR","SERVICE\_USER\_COUNT")  
#plot  
ggplot(SectNTakeTrend,aes(x=as.Date(OCCUPANCY\_DATE),y=SERVICE\_USER\_COUNT, color=SECTOR))+  
 geom\_line()+  
 labs(x="Date",y="Total Intakes",color="Sector")

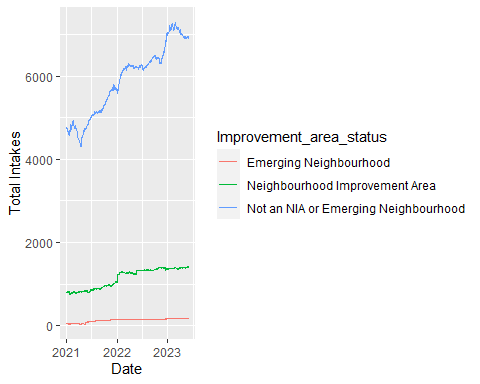


* Mixed Adults consistently have the most intakes however there has been a growing trend in families sector programs.
* All of the trends show random walk qualities.

*Neighbourhood Improvement Areas*

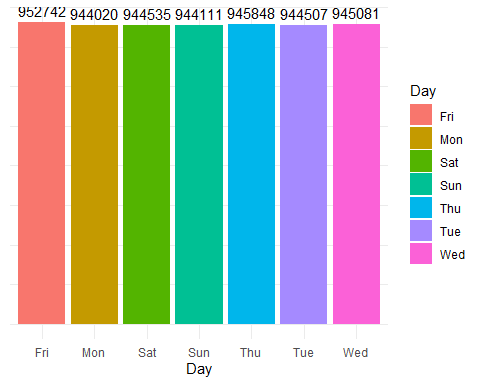
* Neighbourhood Improvement Areas are neighbourhoods that have several inequalities on several indicators of well-being.

##Pivoting  
ImpArea<-as.data.frame(xtabs(data=shltr, shltr$SERVICE\_USER\_COUNT ~ shltr$OCCUPANCY\_DATE + shltr$Improv\_Status\_Area))  
colnames(ImpArea)<-c('OCCUPANCY\_DATE','Improvement\_area\_status','SERVICE\_USER\_COUNT')  
  
#plot  
ggplot(ImpArea,aes(x=as.Date(OCCUPANCY\_DATE), y=SERVICE\_USER\_COUNT,color=Improvement\_area\_status))+  
 geom\_line()+  
 labs(x='Date', y='Total Intakes', color='Improvement\_area\_status')



* We see that the majority of shelter intakes actually come from programs that are not in a Neighbourhood Improvement Area or Emerging Neighbourhood.

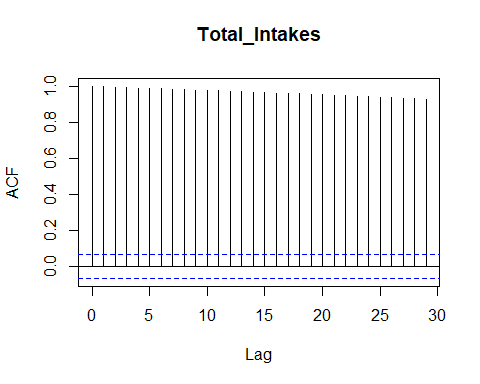
#Day of the Week Analysis   
DayNTake<-aggregate(shltr$SERVICE\_USER\_COUNT ~ weekdays(shltr$OCCUPANCY\_DATE, abbreviate = TRUE ),  
 data = shltr,  
 FUN = sum)  
colnames(DayNTake)<-c('Day','TOTAL\_INTAKES')  
  
#Plot  
ggplot(DayNTake,aes(x=as.character(Day), y = TOTAL\_INTAKES, fill= Day))+  
 geom\_bar(stat='identity')+  
 geom\_text(aes(label=format(TOTAL\_INTAKES,scientific=FALSE)),vjust = -0.5 )+  
 labs(x = 'Day', y=NULL, fill='Day')+  
 scale\_x\_discrete(limits = unique(DayNTake$Day))+  
 theme\_minimal()+  
 theme(axis.text.y = element\_blank(),axis.ticks = element\_blank())



* Looking at the daily breakdown we can see that distribution across the days of the week it is pretty even.

## Correlation

(acf(ts(xtsNTakeShltr)))



##   
## Autocorrelations of series 'ts(xtsNTakeShltr)', by lag  
##   
## 0 1 2 3 4 5 6 7 8 9 10 11 12   
## 1.000 0.998 0.996 0.993 0.991 0.989 0.987 0.984 0.982 0.980 0.977 0.975 0.973   
## 13 14 15 16 17 18 19 20 21 22 23 24 25   
## 0.970 0.968 0.965 0.963 0.960 0.958 0.955 0.953 0.950 0.947 0.945 0.942 0.939   
## 26 27 28 29   
## 0.937 0.934 0.931 0.929

* Understanding the autocorrelation is key to analyzes and assessing the correlation between the time series at specific points and lagged values.
* Looking at the acf plot and outputs we see that there is significance of correlations between lag times 1 - 30. The correlogram shows that all the lags are positively correlated meaning that there is a positive relationship between current observations and lagged values at specific lags. Even though the lags are highly correlated the highest correlated lag is at lag 1. Looking at the acf also shows that there is no seasonality.

## Forecasting

* I am going to forecast the next 30 days using time-series forecast models.
* The models I am going to use and assess are going to be (1) naive (2)SES (3)ARIMA

##Creating test and training sets.  
#For forecasting I will convert my xts object to a ts object.   
tsNTake<-ts(xtsNTakeShltr)  
##Training is all the dates in the time series except the last 30 days (a month)  
trnNTake<-window(tsNTake,end = end(index(tsNTake)[1:(nrow(tsNTake)-31)]))  
##Test  
tstNTake<-window(tsNTake,start=c(end(tsNTake)[1] - 30, end(tsNTake)[2]))  
  
end(tstNTake)

## [1] 883 1

start(tstNTake)

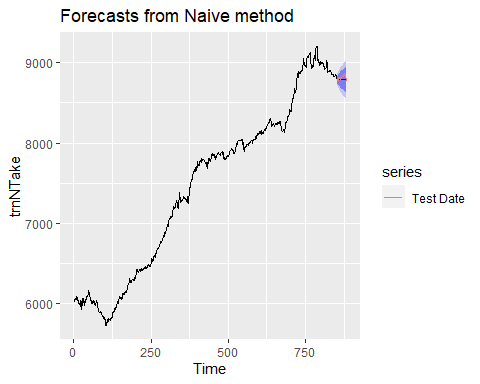
## [1] 853 1

*Naive*

#Naive Test  
library(forecast)  
fcNaiNTake<-naive(trnNTake,h=30)  
end(trnNTake)

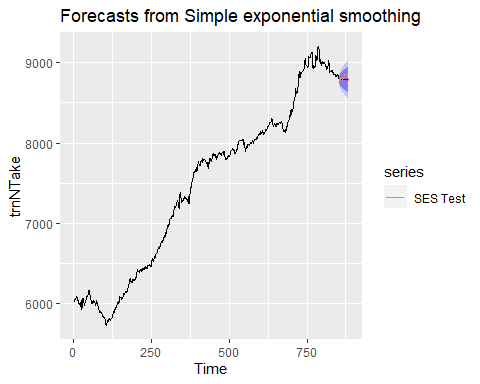
## [1] 852 1

autoplot(fcNaiNTake)+  
 autolayer(tstNTake,series='Test Date')



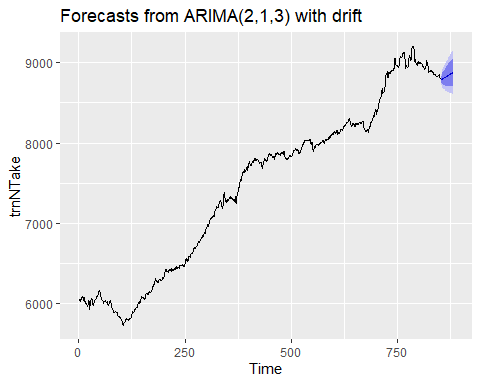
*Simple Exponential Smoothing*

#Simple Exponential Smoothing   
fcSesNTake<-ses(trnNTake,h=30)  
autoplot(fcSesNTake)+  
 autolayer(tstNTake,series = "SES Test")



*ARIMA*

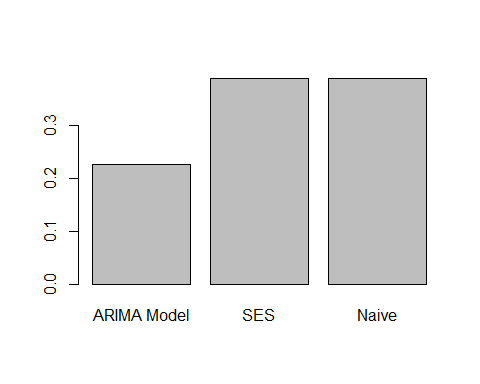
ArNTakefit<-auto.arima(trnNTake)  
fcArNTake<-forecast(ArNTakefit,h=31)  
autoplot(fcArNTake)



* The auto.arima function selects the model given the time series. Here it selected a ARIMA(2,1,3) with a drift. Meaning that data has been differenced with a lag of 1, 2 past observations are regressed and 2 past errors are being used in the equation. Looking at the strong linear trend we see that a 1 difference is good enough to make transform the data to stationary. This difference suits the lag of 1 I noted in the acf function above.

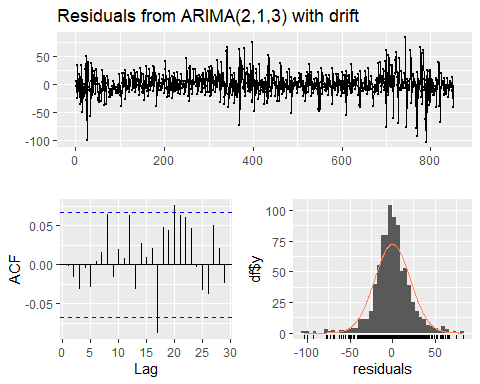
## Evaluating model performance.

MapeARNTake<-accuracy(fcArNTake,tsNTake)[2,5]  
MapeSesNTake<-accuracy(fcSesNTake,tsNTake)[2,5]  
MapeNaiNTake<-accuracy(fcNaiNTake,tsNTake)[2,5]  
  
MapeValues<-c(accuracy(fcArNTake,tsNTake)[2,5],accuracy(fcSesNTake,tsNTake)[2,5],accuracy(fcNaiNTake,tsNTake)[2,5])  
  
barplot(MapeValues,names.arg = c('ARIMA Model','SES','Naive'))  
text(x = 1:length(MapeValues), y = MapeValues + 0.5, labels = MapeValues, pos = 3, cex = 0.8)



* We can compare the accuracy/quality of the forecast using the MAPE test result. The ARIMA model has lowest value with accuracy(fcArNTake,tsNTake)[2,5] suggesting it has the Can compare MAPE, small value indicates a better forecast. We are interested in the test set error measures.
* The MAPE measures the average percentage difference between the predicted values of the forecast and the actual values. This gives us a idea of forecasts error.

checkresiduals(fcArNTake)



##   
## Ljung-Box test  
##   
## data: Residuals from ARIMA(2,1,3) with drift  
## Q\* = 6.2126, df = 5, p-value = 0.2861  
##   
## Model df: 5. Total lags used: 10

* The Ljung-Box test shows no significant autocorrelation. When looking at the ACF plot this is confirmed. The residuals also look like a white noise plot and the histogram shows a normal distribution of the residuals. These qualities indicate that the ARIMA model (2,1,3) is a good quality model for forecasting daily shelter intakes.

## Next Steps

* Below are a list of metrics that will be analyzed for final submission.

##Daily Count of programs operating   
OpenShltr<-aggregate(PROGRAM\_ID ~ OCCUPANCY\_DATE,data = shltr,FUN = length)  
colnames(OpenShltr)[2]<-'TotalOpenPrograms'  
  
##Over capacity programs   
OverCapShltr<-aggregate(OVER\_OCCUPIED ~ OCCUPANCY\_DATE ,data=shltr,FUN=sum)  
colnames(OverCapShltr)[2]<-'TotalOverCapacityShltrs'  
  
#Proportion of shltrs operating at over capacity  
OverCapPropShltr<-left\_join(OpenShltr,OverCapShltr,by = 'OCCUPANCY\_DATE')  
OverCapPropShltr$OverCapRate<-round((OverCapPropShltr$TotalOverCapacityShltrs/OverCapPropShltr$TotalOpenPrograms)\*100,digits = 2)  
OverCapPropShltr<-OverCapPropShltr[,c(1,4)]  
  
#Capacity Rates  
AvgOccPerShltr<-aggregate(OCCUPANCY\_RATE ~ OCCUPANCY\_DATE,data=shltr,FUN=mean)  
AvgOccPerShltr$OCCUPANCY\_RATE<-round(AvgOccPerShltr$OCCUPANCY\_RATE,digits = 2)

* I would like to understand the relationship and correlation between weather and shelter metrics. Below is a script that would bring in weather information to the shelter data source.

#Bring in weather info   
url<-"https://climate.weather.gc.ca/climate\_data/daily\_data\_e.html?hlyRange=2009-12-10%7C2023-05-14&dlyRange=2010-02-02%7C2023-05-13&mlyRange=%7C&StationID=48549&Prov=ON&urlExtension=\_e.html&searchType=stnName&optLimit=yearRange&StartYear=1840&EndYear=2023&selRowPerPage=25&Line=1&searchMethod=contains&txtStationName=Toronto+city+centre&timeframe=2&Day=14&"  
  
##Webscrapping values prior to 2023  
web\_tabledf2022<-NULL  
for (y in 2010:2022){  
 for (m in 1:12) {  
 urlperiod<-paste(url,'Year=',y,'&Month=',m,'#',sep='')  
 #print(urlperiod)  
 webpage<-read\_html(urlperiod)  
 web\_table<- webpage %>% html\_nodes('table')%>%.[1] %>%  
 html\_table() %>% .[[1]]  
 web\_table$Period<-paste(y,'-',m,sep='')  
 web\_table$address<-urlperiod  
 web\_tabledf2022<-rbind(web\_table,web\_tabledf2022)  
 }  
}  
  
  
  
##Websrapping values during 2023  
url<-"https://climate.weather.gc.ca/climate\_data/daily\_data\_e.html?hlyRange=2009-12-10%7C2023-05-14&dlyRange=2010-02-02%7C2023-05-13&mlyRange=%7C&StationID=48549&Prov=ON&urlExtension=\_e.html&searchType=stnName&optLimit=yearRange&StartYear=1840&EndYear=2023&selRowPerPage=25&Line=1&searchMethod=contains&txtStationName=Toronto+city+centre&timeframe=2&Day=14&Year=2023&Month="  
web\_tabledf2023<-NULL  
for (t in 1:6) {  
 urlperiod<-paste(url,t,'#',sep='')  
 #print(urlperiod)  
 webpage<-read\_html(urlperiod)  
 web\_table<- webpage %>% html\_nodes('table')%>%.[1] %>%  
 html\_table() %>% .[[1]]  
 web\_table$Period<-paste('2023-',t,sep='')  
 web\_table$address<-urlperiod  
 web\_tabledf2023<-rbind(web\_table,web\_tabledf2023)  
}  
  
weatherDF<-rbind(web\_tabledf2023,web\_tabledf2022)  
weatherDF<-weatherDF[!is.na(as.numeric(weatherDF$DAY)),]

## Warning in `[.tbl\_df`(weatherDF, !is.na(as.numeric(weatherDF$DAY)), ): NAs  
## introduced by coercion

#Clean date column  
#YYYY-MM-DD  
weatherDF$Full\_Date<-paste(weatherDF$Period,'-',weatherDF$DAY,sep='')  
weatherDF$Full\_Date<-as.Date(weatherDF$Full\_Date)  
##Convert columns to numeric  
weatherDF$`Max Temp Definition°C`<-as.numeric(weatherDF$`Max Temp Definition°C`)

## Warning: NAs introduced by coercion

weatherDF$`Min Temp Definition°C`<-as.numeric(weatherDF$`Min Temp Definition°C`)

## Warning: NAs introduced by coercion

weatherDF$`Mean Temp Definition°C`<-as.numeric(weatherDF$`Mean Temp Definition°C`)

## Warning: NAs introduced by coercion

weatherDF$`Total Precip Definitionmm`<-as.numeric(weatherDF$`Total Precip Definitionmm`)

## Warning: NAs introduced by coercion

##For missing values use the last value  
#Could sort first  
#The na.locf will fill na values with the value before it  
weatherDF$`Total Precip Definitionmm`<-na.locf(weatherDF$`Total Precip Definitionmm`)   
weatherDF$`Max Temp Definition°C`<-na.locf(weatherDF$`Max Temp Definition°C`)  
weatherDF$`Min Temp Definition°C`<-na.locf(weatherDF$`Min Temp Definition°C`)  
weatherDF$`Mean Temp Definition°C`<-na.locf(weatherDF$`Mean Temp Definition°C`)  
  
##Filter the weather data on dates to match the range of dates that are in the shltr dataset.   
weatherDF<-weatherDF%>%filter(between(weatherDF$Full\_Date,min(shltr$OCCUPANCY\_DATE),max(shltr$OCCUPANCY\_DATE)))  
  
shltr<-left\_join(shltr,weatherDF,by=c('OCCUPANCY\_DATE'='Full\_Date'))